Zoea

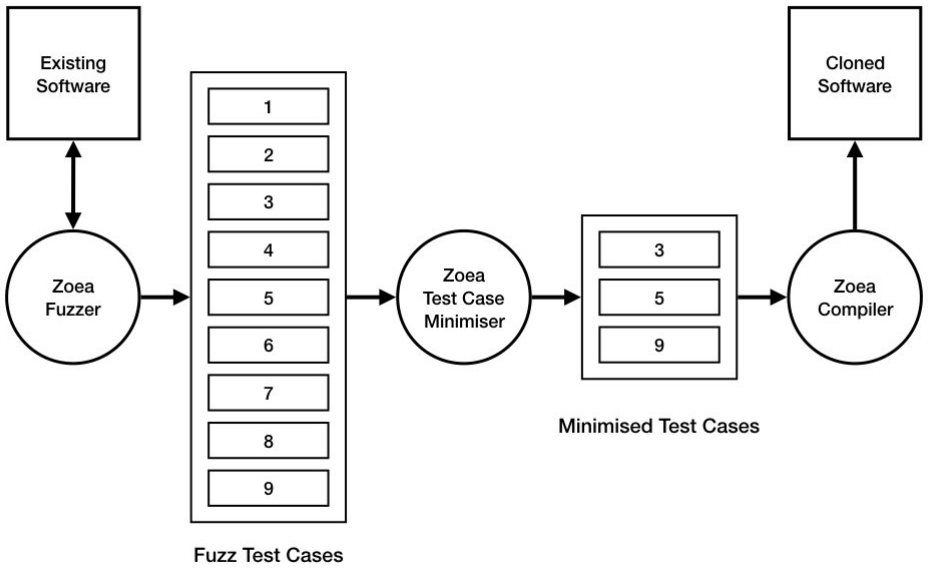
Research Note 5

Migrating Existing Software to Zoea

Edward McDaid & Sarah McDaid
17 Jul 2020

Zoea is the first AI that can automatically generate software of any size from a set of test cases. This is a big deal that will ultimately transform the nature of software development forever. But what about all the software in other languages that already exists?

Migrating existing software to Zoea



One thing that people are going to want to do is to migrate their legacy systems to Zoea. This will be much easier if the conversion process is completely automated.

Zoea enables many existing programs to be automatically converted to Zoea code. This is regardless of the programming language used in the original software. Migration is accomplished through a combination of fuzzing and test case minimisation. The migration process only requires access to an executable version of the software and does not need the source code for the original program.

Programs in Zoea are basically a set of test cases that describe the required behaviour. As a result we can recreate any program in Zoea if we can produce the corresponding set of test cases. The fastest way to produce a lot of test cases with very little effort is through a process called 'fuzzing'.

Fuzzing involves producing lots of random inputs that are fed into a running copy of the program. As the inputs are random many of them will not work correctly. This doesn't really matter. Where a set of inputs do cause the program to produce an output we have a potential step in a test case. This process is normally very fast and doesn't involve much effort or expense so we can literally try millions of different input combinations. Eventually we will have a large number of test cases that cover every code path in the original program.

At this stage we could just feed the generated test cases into the Zoea compiler and it would eventually produce the required program. However, the test cases produced by the fuzzer include a lot of redundancy in the sense that many test cases correspond to the same code path. We can speed up compilation significantly by removing redundant test cases through a process called test case minimisation.

Test case minimisation is very straightforward in Zoea. When Zoea identifies a candidate code fragment for a given test case it tries that solution against all the other test cases to see whether it works for them also. At each step only the test case that gives rise to a new case solution is retained and all of the rest that correspond to the same solution are discarded. In the end we are left with a much smaller subset of test cases that completely cover the code paths for the original program. This can then be fed into the Zoea compiler to produce the equivalent program. Further testing is always recommended to ensure that the behaviour of the new program is correct and complete.

Naturally the overall time and resources required for migration is dependant on the size of the original program. However as with the Zoea compiler both fuzzing and test case minimisation can be parallelised to reduce the time required.

Zoea is the simplest way to develop new software and it also provides effortless migration of legacy code in any language.

Learn more at **zoea.co.uk**